A REPORT

ON

FACE RECOGNITION SYSTEM

BY

SANJEEV MALLICK 2021A7PS2217P

AT

Northcorp Software Private Ltd.

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

JUNE,2023

A REPORT

ON

FACE RECOGNITION SYSTEM

BY

SANJEEV MALLICK 2021A7PS2217P BE (Hons) Computer Science

Prepared in partial fulfillment of the

Practice School-I Course Nos.

BITS C221/BITS C231/BITS C241

AT

Northcorp Software Private Ltd.

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

JUNE,2023

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to the following individuals and organizations for their invaluable support and contributions to this mid-semester report:

1. **Prof. Sunny Kumar Singh**: I am deeply grateful for his guidance, mentorship, and constant encouragement throughout this project. His expertise and insights have been instrumental in shaping the direction of my work.

2. **Northcorp Software Pvt. Ltd**: I extend my heartfelt thanks to the entire team at Northcorp for allowing me to undertake this internship and work on this project. Their support, resources, and collaborative environment have been crucial to my learning and progress.

3. **Colleagues and Peers**: I would like to thank my colleagues and peers at Northcorp for their assistance, discussions, and shared knowledge. These discussions have been invaluable in refining my understanding of the subject matter and enhancing the quality of my work.

4. **Friends and Family**: I am grateful to my friends and family for their unwavering support, encouragement, and understanding throughout this endeavor. Their belief in me has been a constant source of motivation, and I am truly fortunate to have them by my side.

5. **BITS Pilani**: I would like to acknowledge the educational institution, BITS Pilani, for providing me with a solid Computer Science and Engineering foundation. The knowledge and skills acquired during my studies have been instrumental in tackling the challenges of this internship.

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI

(RAJASTHAN)

Practice School Division

Station: **Northcorp Software Private Ltd., Gurugram** Centre: Online

Duration: 50 Days Date of Start: **31/05/2023**

Date of Submission: **19/07/2023**

Title of the Project: **Face Recognition System**

ID No./Name(s)/ Discipline(s)/of the student(s):

2021A7PS2217P	**SANJEEV MALLICK**	**B.E.(Hons) Computer Science**

Name(s) and designation(s) of the expert(s):

Mr. ROHIT PANDEY, (SENIOR MEMBER AND BUSINESS ANALYST AT NORTHCORP PVT. LTD.)

Mr. RISHI SHUKLA, (SENIOR MEMBER AND PRODUCT MANAGER AT BORTHCORP PVT. LTD.)

Name(s) of the PS Faculty:

Dr. SUNNY KUMAR SINGH

Key Words: Python, Computer Vision, Machine Learning, Artificial Intelligence, Face Recognition, Python libraries (Numpy, Pandas), MySQL

Project Areas: Artificial Intelligence and Computer Vision

ABSTRACT

This initiative aims to develop a robust and effective system for autonomously detecting and identifying individual features within a group photograph. The objective is to surmount obstacles such as illumination conditions, facial expressions, and pose, which can make face recognition a difficult endeavour. Utilising OpenCV for image processing tasks such as face detection, Haar cascade features, and image manipulation is part of the methodology. The Face-Recognition library is used to derive facial attributes and compare them with pre-existing face data to identify each detected face.

To enhance the user experience and provide an intuitive interface, the Tkinter library (of python) is utilized to create a graphical user interface. This allows users to easily interact with the system, upload group photos, and obtain accurate face recognition results.

Additionally, the NumPy and Pandas libraries are used for efficient data manipulation and storage of face-related data, facilitating the storage and retrieval of face features and identities.

The conducted experiments demonstrate the efficacy of the proposed method, obtaining high accuracy in detecting and recognising features in group photographs. The system demonstrates to be resilient in the face of a variety of challenges, delivering reliable results even in complex situations.

This project uses the concepts of **Python programming language**, **OpenCV** (Open Computer Vision Library), **Tkinter** (GUI), python libraries which include **Numpy**, **Pandas** and **Face-Recognition and MySQL** for database.

Signature(s) of Student(s):

SANJEEV MALLICK 2021A7PS2217P Sanjeev Mallick

Date: 19 July 2023

Signature of PS Faculty: Date:

TABLE OF CONTENTS

INTRODUCTION

PROBLEM STATEMENT: We are given a dataset of images. Now, we are also given group photos. Our task is to find/identify the individual identities from the group photo using the given photos in the database.

SOLUTION: We can solve this problem by using the concept of Face Detection, Face Recognition and computer vision algorithms. We also have to store the images in a real-time database as well as output the results using database.

DEFINITIONS:

1.Face Detection: Face detection is a computer vision technology that assists in locating and visualising human features in digital images. This technique is a particular application of object detection technology that detects instances of semantic objects of a certain class (such as humans, structures, or automobiles) in digital images and videos. Face recognition has become increasingly important in recent years, particularly in photography, security, and marketing. Face recognition is distinct from Face detection. Face detection detects the presence of features in an image, whereas facial recognition identifies the individual whose face is being detected.

2.Face Recognition: Face recognition is the process of telling whether an item that has already been found is a known or new face. Face detection and face recognition are often mixed with each other. Face Recognition, on the other hand, is used to figure out if the face is a known person or not. To do this, a library of faces is used to check if this face is real.

MOTIVATION AND USEFULNESS BEHIND THE PROJECT:

1. **Identification of Criminals**: Face recognition technology can help catch criminals. The database has all the data on the criminals or gangsters, which is quite helpful in locating them. If system recognize that by utilizing some algorithms to match a specific person with the database. This recognition technology then prevents the crime from happening.

2. **Attendance record**: A face recognition system is helpful for keeping track of student's attendance in schools and colleges so that malicious behavior can be avoided. In order to track employee attendance, offices and colleges use biometric systems. The face recognition approach is also employed by hospitals, colleges, and offices to track staff attendance.

3. **Bank Security Facilities**: Bank uses artificial intelligence to identify any suspect individuals as one of the security measures. This face recognition technique is most helpful in preventing bank fraud.

4. **E-Transactions**: The intelligence services engage in secure electronic exchanges. Nowadays, we can unlock accounts using our faces. Therefore, it will improve transaction security. Transactions only take place when a customer's face is matched with a database.

5. **Airport Security**: In several nations, airports use artificial intelligence-enhanced face detection and identification technology to identify the faces of travelers. It can stop any malicious activity at the airport. The information obtained by using facial recognition is accurate. It avoids any potentially dangerous situations as a result.

6. **Improved User Experience**: Face recognition systems can provide a convenient user experience by eliminating the need for traditional authentication methods such as passwords, PINs, or physical tokens. Users can simply present their faces to gain access to devices, systems, or services, saving time and effort.

It is important to note that while face recognition systems offer numerous benefits, their deployment should be accompanied by proper safeguards to address privacy concerns and ensure ethical and responsible use of the technology.

TECHSTACKS USED IN THE PROJECT:

- **Python Programming Language**
- **Computer Vision**
- **Numpy**
- **Pandas**
- **Face Recognition algorithms**
- **Machine Learning**
- **Artificial Intelligence**
- **Face Detection algorithms**
- **MySQL**
- **Database Management Systems(DBMS)**
- **Flutter and App Development**

Non-Technical Skills inculcated in the project-

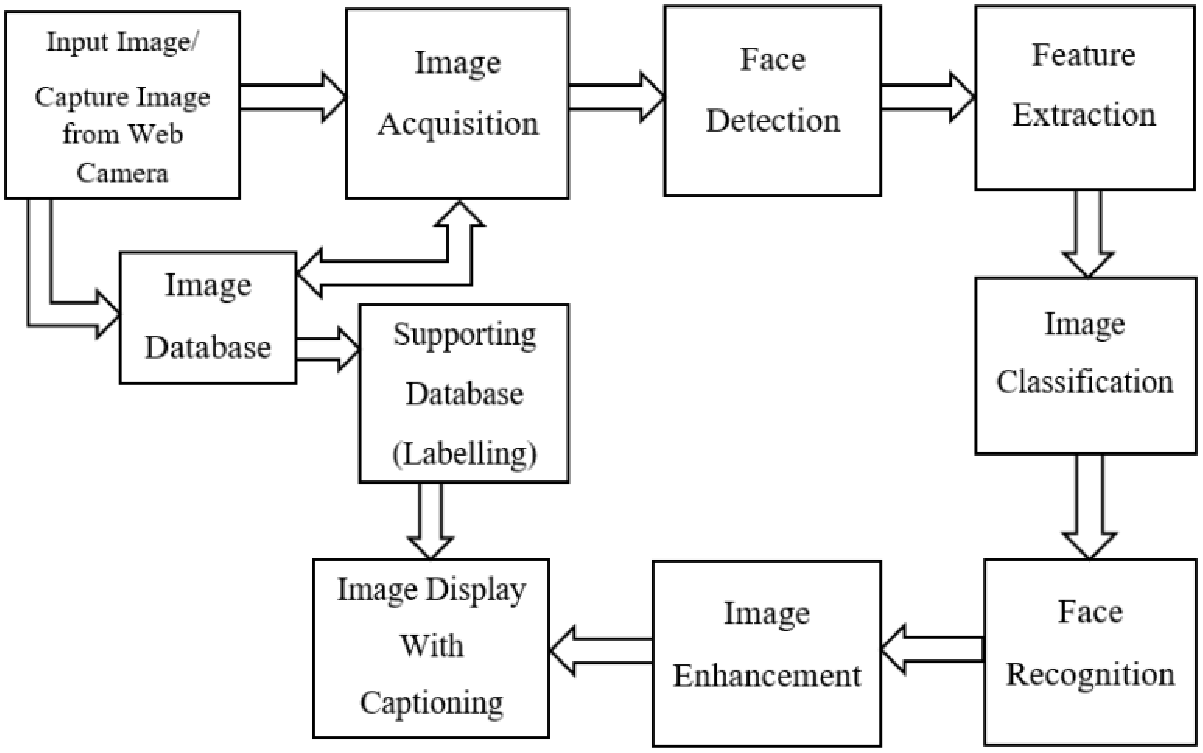
- **Presentations**
- **Microsoft Word**
- **Microsoft PowerPoint**
- **Group Discussion**
- **Graphic Designing**
- **Canva**
- **Figma**
- **Industry experience**

MAIN BODY

METHODOLOGY:

The idea is based on software that works in real time. This is built on a real-time database. So, we use pictures or photos of people to help us figure out who they are. A dataset is a group of photos of different people that are used for data training. Data training is very important for identification and face recognition.

First, input an image from a directory for testing and detection. This is equivalent to reading an image from the user's gallery or taking a photograph with a webcam or external camera. Then, compute the requisite detection features for a specific image, such as edge, line, and center-surrounded features. Haar Cascade Classifier is utilized for face detection.



(The above picture shows the method of face-detection and face-recognition)

STEPS:

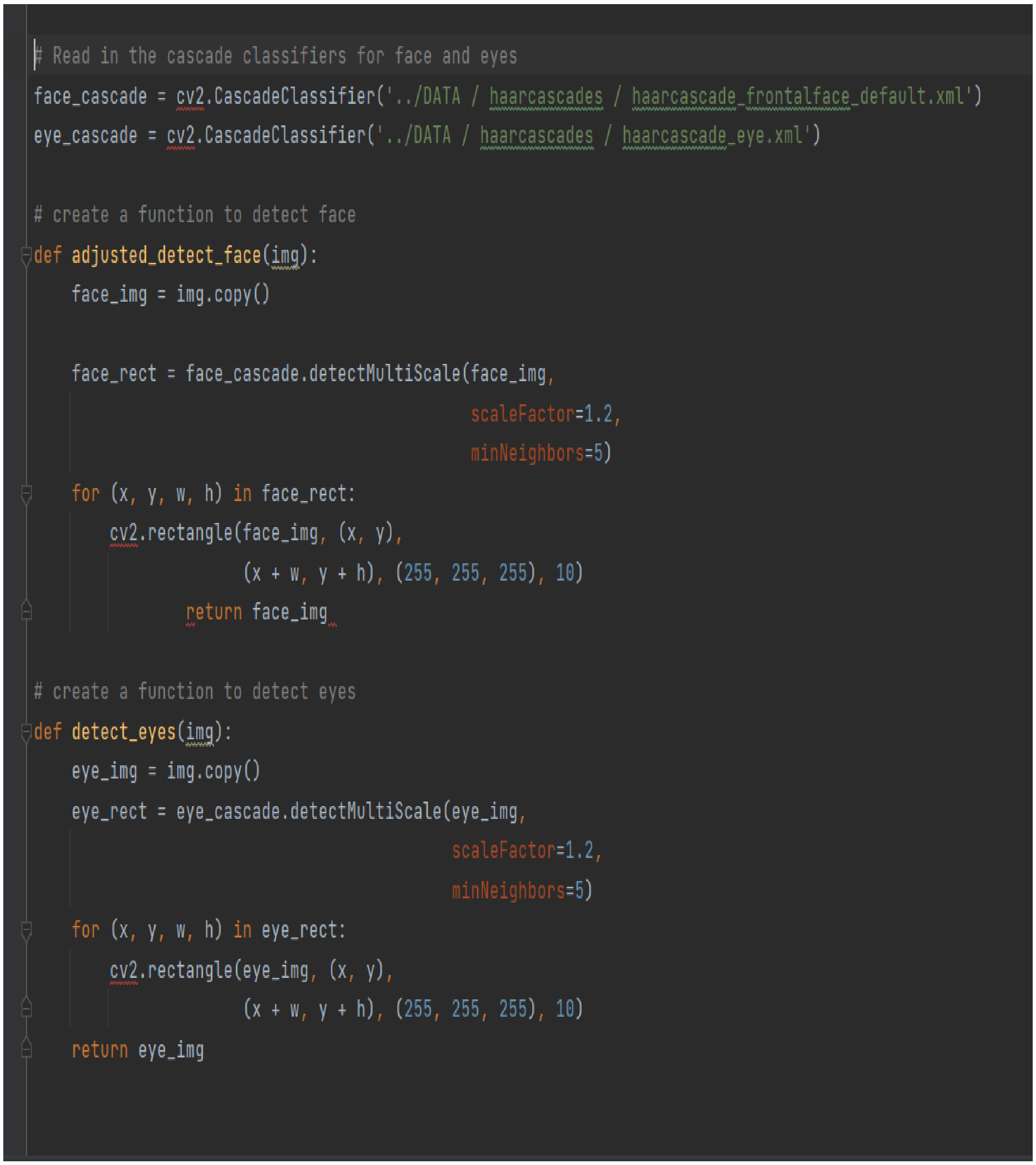
1. **Image Database**: Save around 100 photos of individuals in database. From this database of images only, we have to identify individuals in the group photos. Faces of person present in group photo which matches with all these photos will be identified or recognized successfully.

2. **Face Detection using Python-OpenCV:** Face detection is merely a component of computer technology included in extensive networking. The method is used to recognize human features from video frames and also from tested images. Deep learning is merely a subset of the machine learning algorithm based on the artificial intelligence (AI) technique, which is a network-based structure that operates on the face using neural networks. Using various algorithms, if a human is present in a particular photograph, only that human's face will be detected; non-human faces will not be detected.

3. **Face Identification using Python-OpenCV:** Various algorithms support face recognition. Face detection is crucial to facial recognition. Face must therefore first be detected prior to recognition. Face detection displays the sizes and locations of features in group photographs. Face detection is performed prior to digital image recognition for the purpose of identifying and isolating faces. In this project, we will use Haar Cascade Algorithm for face recognition.

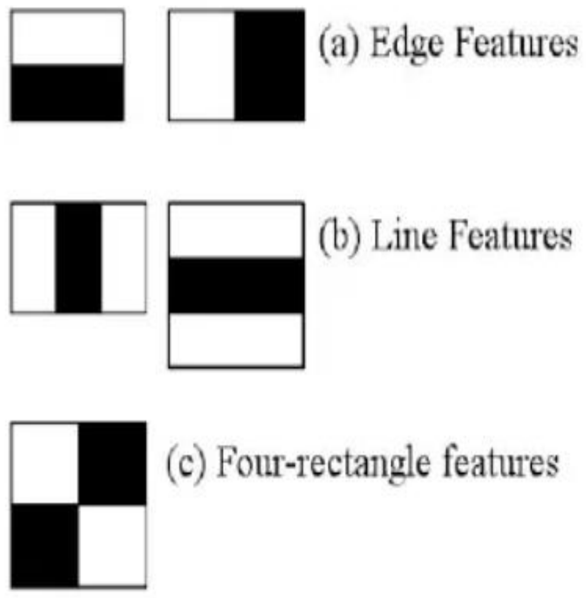
   Haar Cascade Algorithm: The Haar cascade algorithm is a machine learning-based approach used for object detection in images. The algorithm works by using a set of pre-defined Haar-like features to identify regions of interest within an image. These features are simple rectangular patterns that capture certain visual characteristics like edges, lines, or corners. Each feature represents the difference in pixel intensities between adjacent rectangular regions.

Haar Cascade Feature Implementation:



```python
# Read in the cascade classifiers for face and eyes
face_cascade = cv2.CascadeClassifier('../DATA / haarcascades / haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('../DATA / haarcascades / haarcascade_eye.xml')

# create a function to detect face
def adjusted_detect_face(img):
    face_img = img.copy()

    face_rect = face_cascade.detectMultiScale(face_img,
                                              scaleFactor=1.2,
                                              minNeighbors=5)
    for (x, y, w, h) in face_rect:
        cv2.rectangle(face_img, (x, y),
                      (x + w, y + h), (255, 255, 255), 10)
            return face_img

# create a function to detect eyes
def detect_eyes(img):
    eye_img = img.copy()
    eye_rect = eye_cascade.detectMultiScale(eye_img,
                                            scaleFactor=1.2,
                                            minNeighbors=5)
    for (x, y, w, h) in eye_rect:
        cv2.rectangle(eye_img, (x, y),
                      (x + w, y + h), (255, 255, 255), 10)
    return eye_img
```

- The first step is to collect the Haar features. A Haar feature is essentially calculations that are performed on adjacent rectangular regions at a specific location in a detection window. The calculation involves summing the pixel intensities in each region and calculating the differences between the sums. Here are some examples of Haar features below.



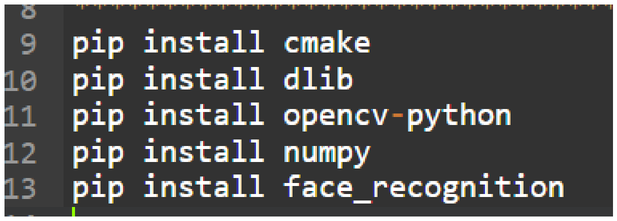
Types of Haar features. (Image Source)

Haar cascades are one of many algorithms that are currently being used for object detection. One thing to note about Haar cascades is that it is very important to reduce the false negative rate, so make sure to tune hyperparameters accordingly when training your model.

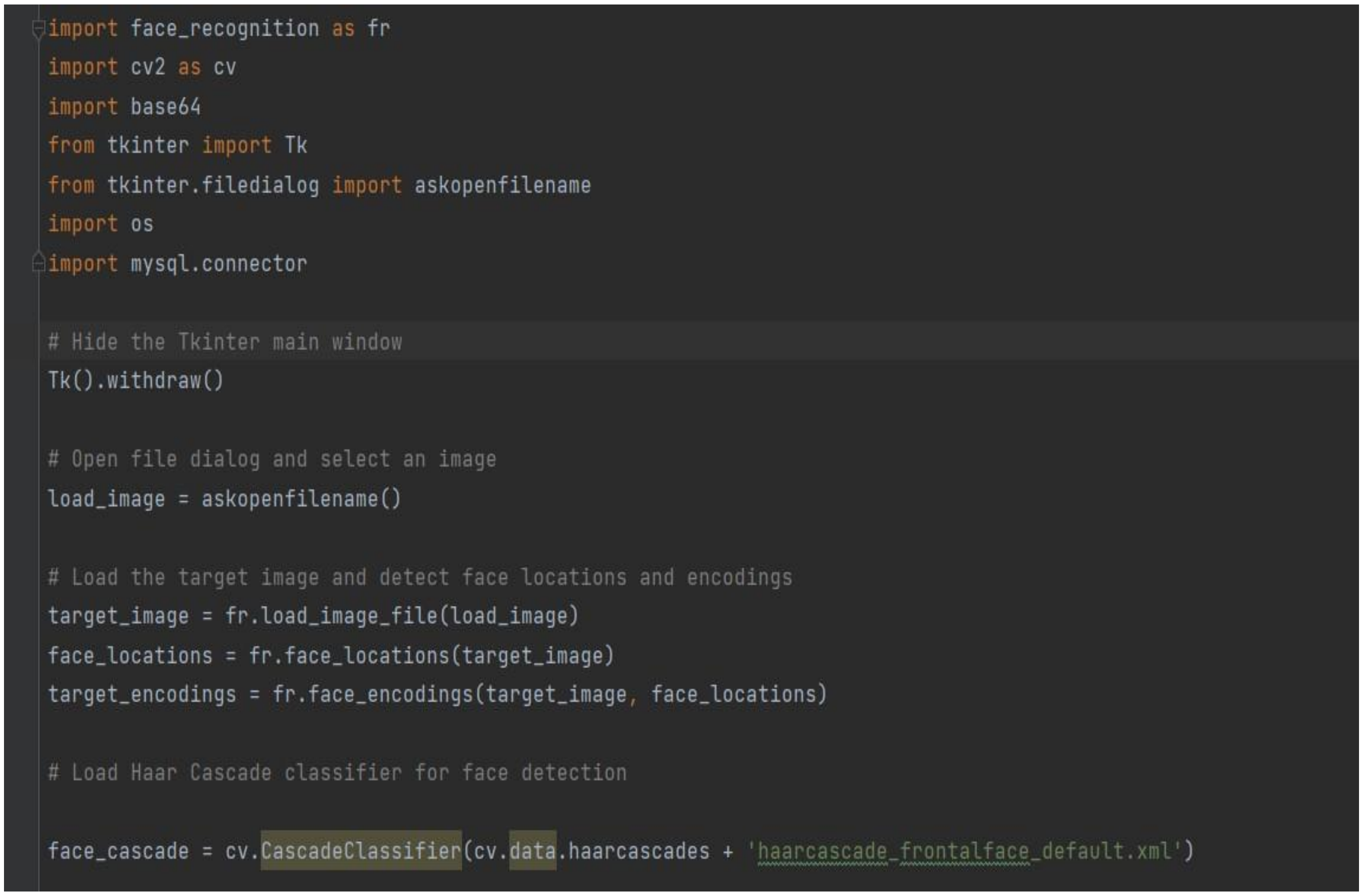
WORKING AND IMPLEMENTATION:

Here is the screenshot of the code below to implement the task.

- Installing the essential python libraries



```
pip install cmake
pip install dlib
pip install opencv-python
pip install numpy
pip install face_recognition
```



```
import face_recognition as fr
import cv2 as cv
import base64
from tkinter import Tk
from tkinter.filedialog import askopenfilename
import os
import mysql.connector

# Hide the Tkinter main window
Tk().withdraw()

# Open file dialog and select an image
load_image = askopenfilename()

# Load the target image and detect face locations and encodings
target_image = fr.load_image_file(load_image)
face_locations = fr.face_locations(target_image)
target_encodings = fr.face_encodings(target_image, face_locations)

# Load Haar Cascade classifier for face detection

face_cascade = cv.CascadeClassifier(cv.data.haarcascades + 'haarcascade_frontalface_default.xml')
```

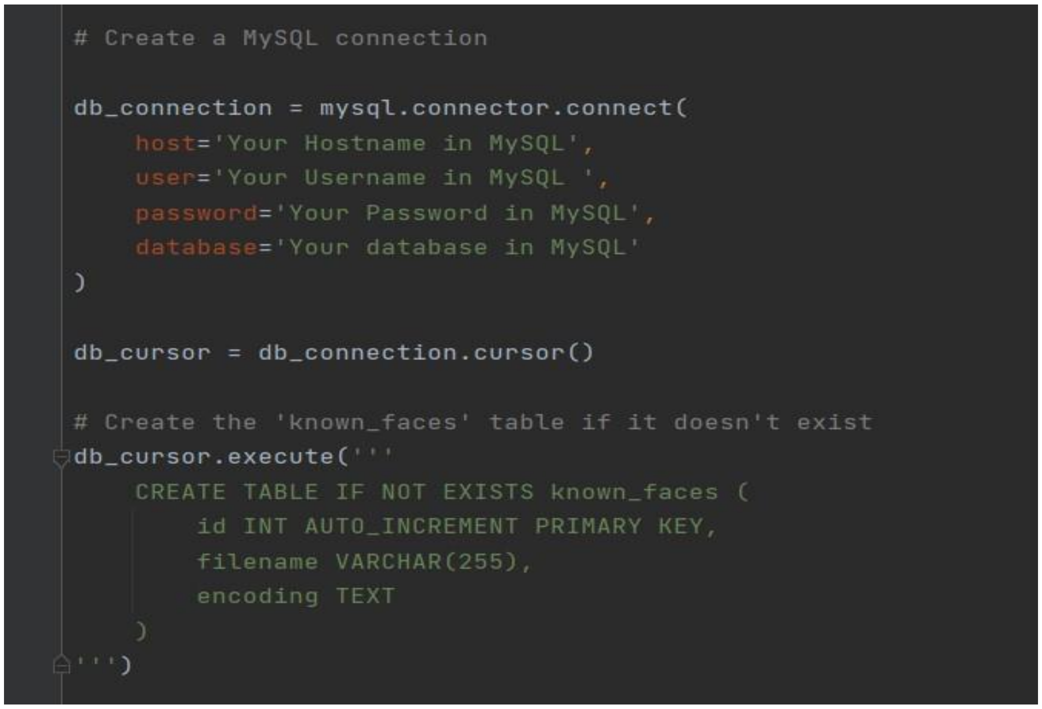
- **Import necessary modules:** The script begins by importing the required libraries/modules for the project, including face_recognition, OpenCV (cv2), base64, Tkinter, and mysql.connector.

- **Hide Tkinter main window:** Tkinter is a GUI library, and by using Tk().withdraw(), it hides the main window that would otherwise appear when using Tkinter.

- **Open file dialog and select an image:** The script opens a file dialog box using askopenfilename() from Tkinter, allowing the user to choose an image file. The selected image file is then stored in the variable load_image.

- **Load the target image and detect face locations and encodings:** The selected target image is loaded using face_recognition's load_image_file() function. Then, the script uses face_recognition.face_locations() and face_recognition.face_encodings() to detect face locations and encodings within the image, respectively. The face locations and encodings are stored in face_locations and target_encodings, respectively.

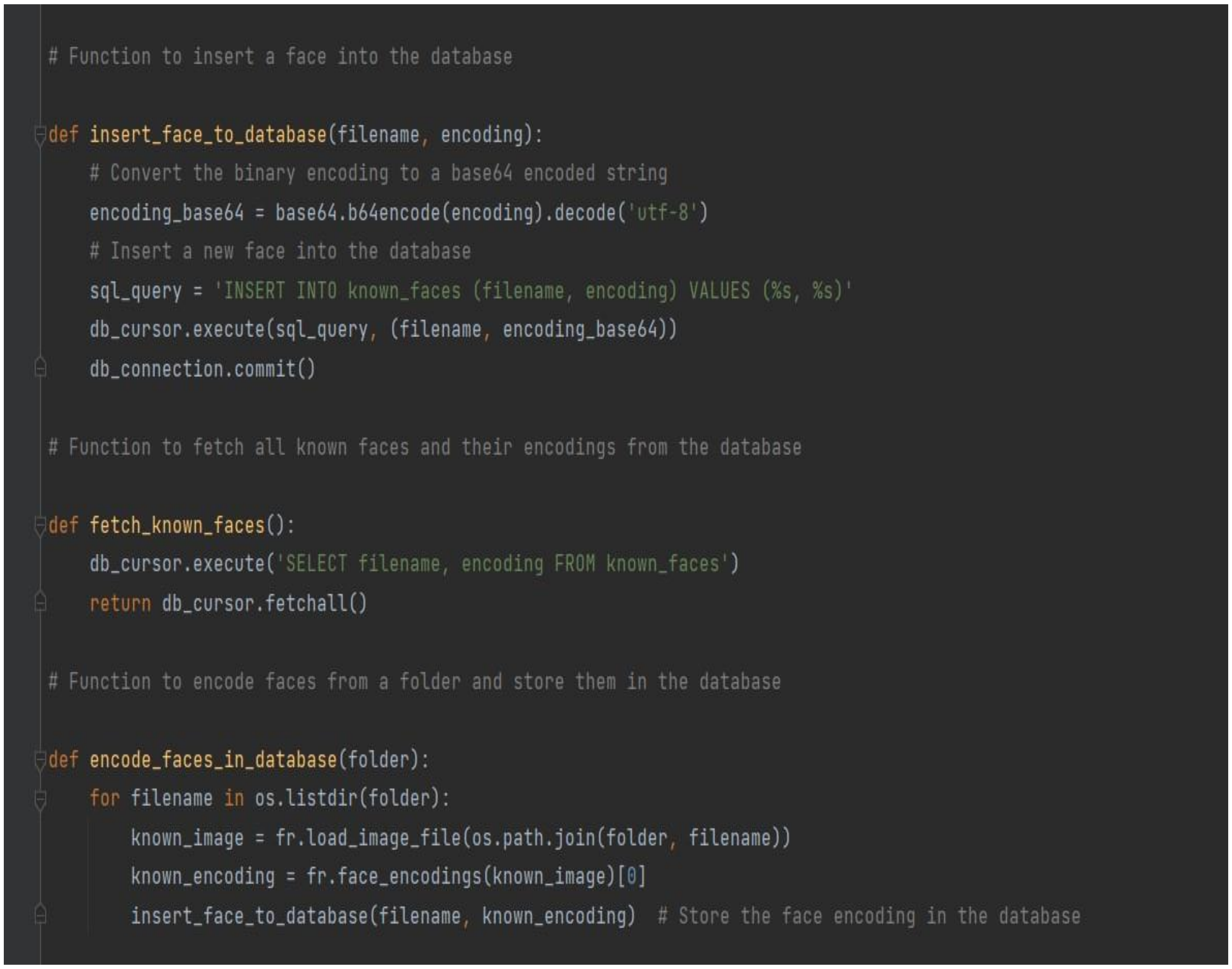
- **Load Haar Cascade classifier for face detection:** The Haar Cascade classifier for face detection from OpenCV is loaded. It will be used later in the code.



```
# Create a MySQL connection

db_connection = mysql.connector.connect(
    host='Your Hostname in MySQL',
    user='Your Username in MySQL ',
    password='Your Password in MySQL',
    database='Your database in MySQL'
)

db_cursor = db_connection.cursor()

# Create the 'known_faces' table if it doesn't exist
db_cursor.execute('''
    CREATE TABLE IF NOT EXISTS known_faces (
        id INT AUTO_INCREMENT PRIMARY KEY,
        filename VARCHAR(255),
        encoding TEXT
    )
''')
```

- **Create a MySQL connection:** The script establishes a connection to a MySQL database using the mysql.connector.connect() function. The appropriate database credentials, such as hostname, username, password, and database name, are provided in the function.

- **Create the 'known_faces' table if it doesn't exist:** The script executes a SQL query to create a table named 'known_faces' in the database if it doesn't already exist. The table has three columns: 'id', 'filename', and 'encoding'.



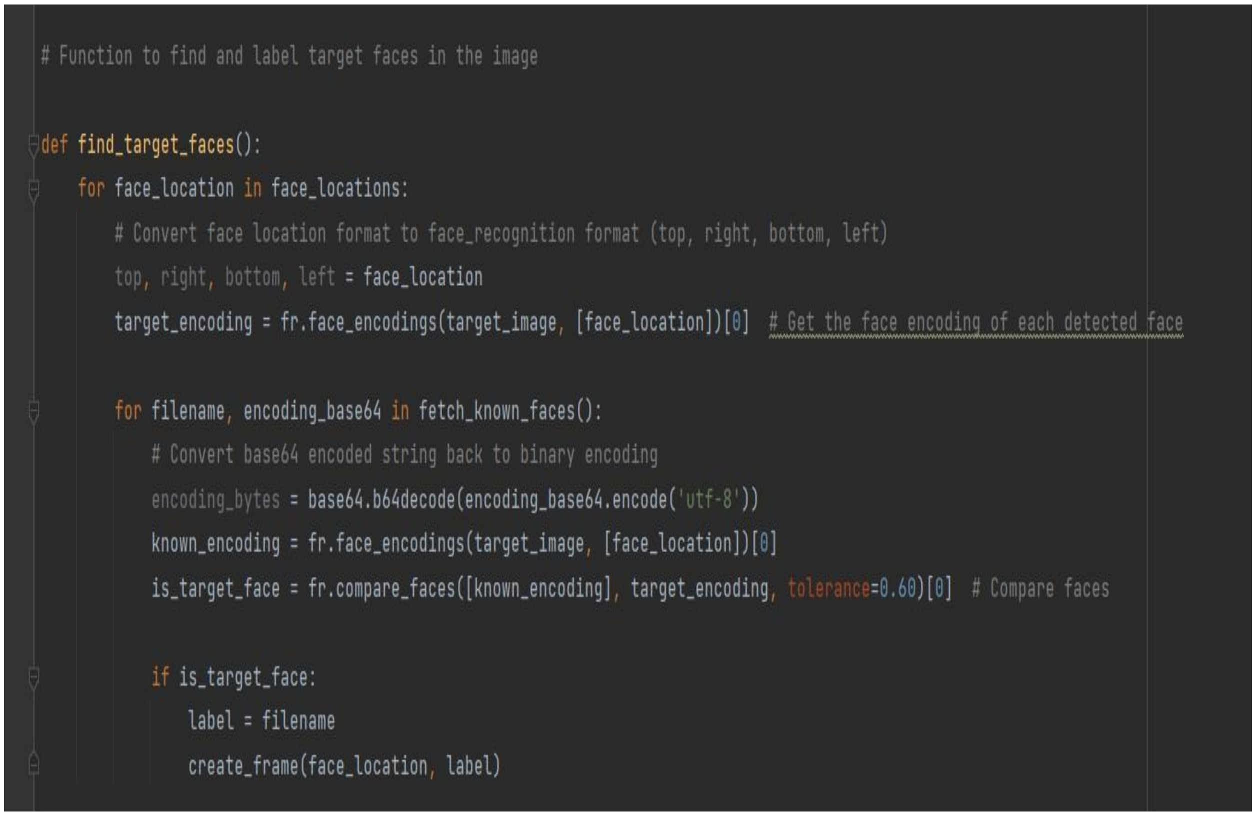
```
# Function to insert a face into the database

def insert_face_to_database(filename, encoding):
    # Convert the binary encoding to a base64 encoded string
    encoding_base64 = base64.b64encode(encoding).decode('utf-8')
    # Insert a new face into the database
    sql_query = 'INSERT INTO known_faces (filename, encoding) VALUES (%s, %s)'
    db_cursor.execute(sql_query, (filename, encoding_base64))
    db_connection.commit()

# Function to fetch all known faces and their encodings from the database

def fetch_known_faces():
    db_cursor.execute('SELECT filename, encoding FROM known_faces')
    return db_cursor.fetchall()

# Function to encode faces from a folder and store them in the database

def encode_faces_in_database(folder):
    for filename in os.listdir(folder):
        known_image = fr.load_image_file(os.path.join(folder, filename))
        known_encoding = fr.face_encodings(known_image)[0]
        insert_face_to_database(filename, known_encoding)  # Store the face encoding in the database
```

Define functions to work with the database:

- **insert_face_to_database():** This function is used to insert a face into the database. The face's filename and encoding are provided as parameters, and the encoding is first converted to a base64 encoded string before insertion.

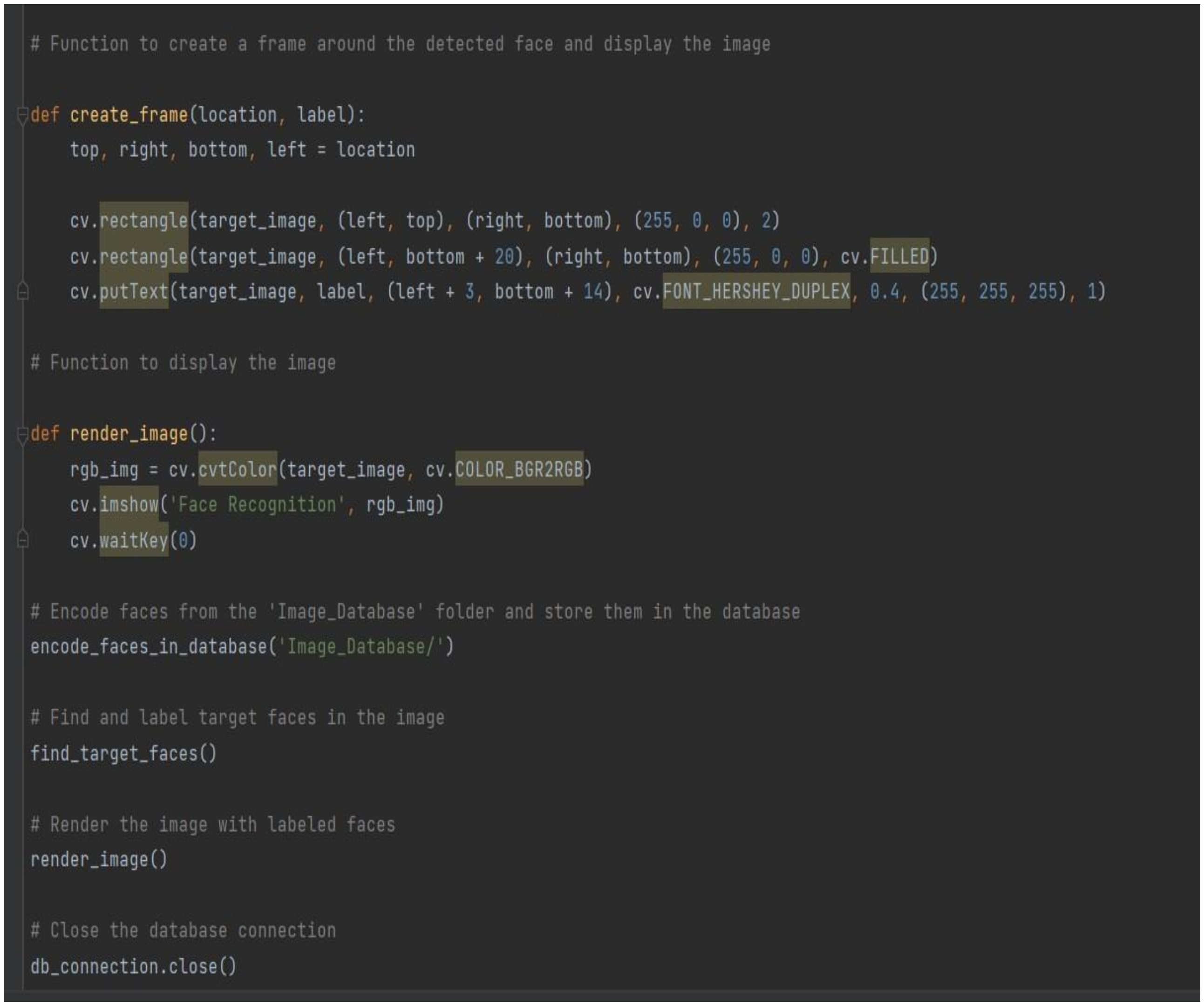
- **fetch_known_faces():** This function fetches all known faces and their encodings from the database.

- **encode_faces_in_database():** This function encodes faces from a specified folder and stores them in the database. It loops through all the files in the folder, loads each image, and extracts the face encoding using face_recognition. Then, it calls insert_face_to_database() to store the face encoding in the database.



```python
# Function to find and label target faces in the image

def find_target_faces():
    for face_location in face_locations:
        # Convert face location format to face_recognition format (top, right, bottom, left)
        top, right, bottom, left = face_location
        target_encoding = fr.face_encodings(target_image, [face_location])[0]  # Get the face encoding of each detected face

        for filename, encoding_base64 in fetch_known_faces():
            # Convert base64 encoded string back to binary encoding
            encoding_bytes = base64.b64decode(encoding_base64.encode('utf-8'))
            known_encoding = fr.face_encodings(target_image, [face_location])[0]
            is_target_face = fr.compare_faces([known_encoding], target_encoding, tolerance=0.60)[0]  # Compare faces

            if is_target_face:
                label = filename
                create_frame(face_location, label)
```

- **Find and label target faces in the image**: The script loops through the detected face locations (from step 4) using the face_locations list. For each face location, it converts the location format to the face_recognition format. Then, it retrieves the face encoding of each detected face using fr.face_encodings() and compares it with known faces' encodings fetched from the database. The comparison is done using fr.compare_faces() with a specified tolerance of 0.60.



```python
# Function to create a frame around the detected face and display the image

def create_frame(location, label):
    top, right, bottom, left = location

    cv.rectangle(target_image, (left, top), (right, bottom), (255, 0, 0), 2)
    cv.rectangle(target_image, (left, bottom + 20), (right, bottom), (255, 0, 0), cv.FILLED)
    cv.putText(target_image, label, (left + 3, bottom + 14), cv.FONT_HERSHEY_DUPLEX, 0.4, (255, 255, 255), 1)

# Function to display the image

def render_image():
    rgb_img = cv.cvtColor(target_image, cv.COLOR_BGR2RGB)
    cv.imshow('Face Recognition', rgb_img)
    cv.waitKey(0)

# Encode faces from the 'Image_Database' folder and store them in the database
encode_faces_in_database('Image_Database/')

# Find and label target faces in the image
find_target_faces()

# Render the image with labeled faces
render_image()

# Close the database connection
db_connection.close()
```

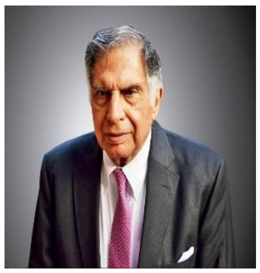
- Create a frame around the detected face and display the image: For each target face detected in step 9, the script creates a rectangle around the face and adds a label (filename) to it using OpenCV's rectangle() and putText() functions.

- Display the image with labeled faces: The labeled image is displayed using OpenCV's imshow() function.

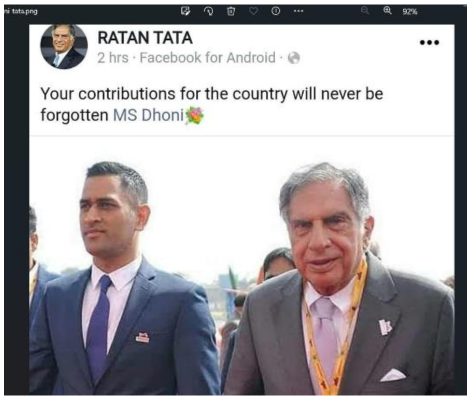
- Close the database connection: After all operations are complete, the script closes the MySQL database connection.

WORKING:

(These are the sample input images for detection and recognition)



Narendra Modi **Sundar Pichai** **Ratan Tata** **Rishi Sunak** **MS Dhoni**

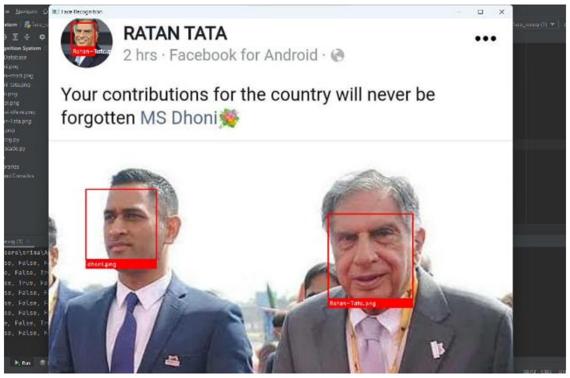


RATAN TATA
2 hrs · Facebook for Android ·

Your contributions for the country will never be forgotten MS Dhoni💐

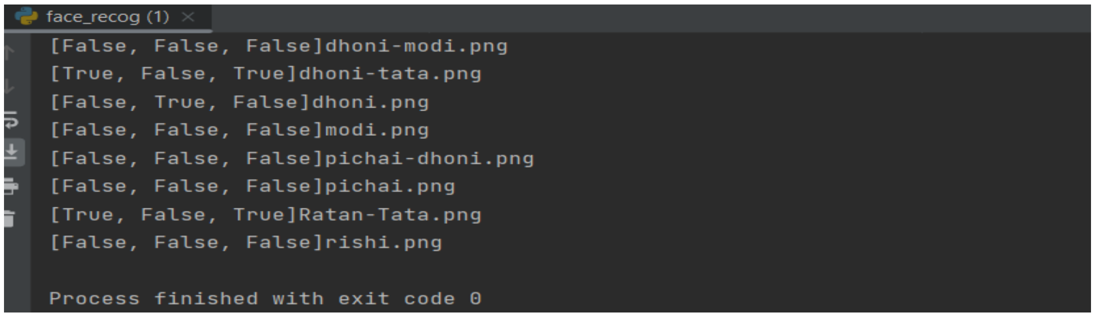
(The above picture has MS Dhoni in left and Ratan Tata in right).

If we try to run the task, then the following happens



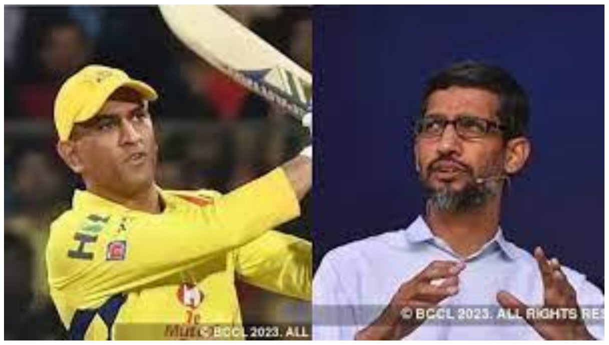
RATAN TATA
2 hrs · Facebook for Android ·
Your contributions for the country will never be forgotten MS Dhoni
dhoni.png
Ratan-Tata.png

Output in the terminal:



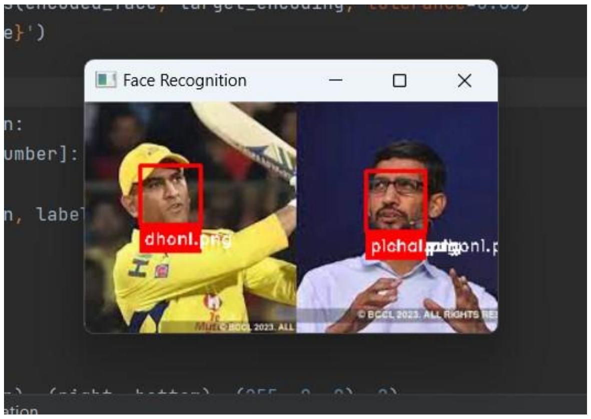
```
face_recog (1)
[False, False, False]dhoni-modi.png
[True, False, True]dhoni-tata.png
[False, True, False]dhoni.png
[False, False, False]modi.png
[False, False, False]pichai-dhoni.png
[False, False, False]pichai.png
[True, False, True]Ratan-Tata.png
[False, False, False]rishi.png

Process finished with exit code 0
```

Now, if we consider another image for face detection and recognition

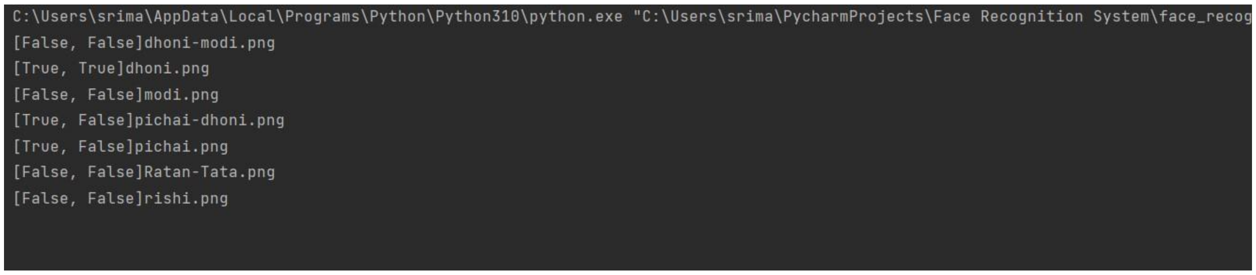


(The above picture has MS Dhoni in left and Sundar Pichai in right).

If we try to run the task, then the following happens



Output in the terminal:



```
C:\Users\srima\AppData\Local\Programs\Python\Python310\python.exe "C:\Users\srima\PycharmProjects\Face Recognition System\face_recog
[False, False]dhoni-modi.png
[True, True]dhoni.png
[False, False]modi.png
[True, False]pichai-dhoni.png
[True, False]pichai.png
[False, False]Ratan-Tata.png
[False, False]rishi.png
```

Hence, we can see that we can extract individual names which are stored in our database to recognize them in group photos using the concept of Face-Detection and Face-recognition.

CONCLUSION

In conclusion, the development and implementation of a face recognition system for group photos using Python, OpenCV, Tkinter, NumPy, Pandas, and Face-Recognition libraries have proven to be successful and effective. The system has demonstrated its ability to accurately detect and recognize individual faces within a group photo, overcoming challenges such as variations in lighting, expressions, and pose.

Overall, the system has demonstrated high accuracy in recognizing faces from group photos, making it a valuable tool in various applications, including security, social media, and human-computer interaction. The successful implementation of this system opens up possibilities for improving efficiency and accuracy in face recognition tasks, contributing to advancements in the field of computer vision.

Moreover, in the realm of social media, the system can play a vital role in enhancing user experience. By automatically recognizing and tagging individuals in group photos, it simplifies the process of sharing and organizing images, thus creating a more engaging and interactive platform. Additionally, the successful development of this face recognition system also has implications for human-computer interaction. It can be integrated into applications to offer personalized experiences, such as customizing content or functionalities based on recognized users' preferences.

In conclusion, the successful implementation of the face recognition system for group photos represents a significant milestone in computer vision and artificial intelligence. The system's ability to accurately identify faces in challenging environments has practical applications in security, social media, and human-computer interaction. As the field of computer vision continues to evolve, this achievement serves as a stepping stone towards creating even more advanced and socially responsible face recognition systems.

REFERENCES

1. https://en.wikipedia.org/wiki/Face_detection
2. https://www.researchgate.net/publication/326667118_Face_Detection_Techniques_A_Review
3. https://docs.python.org/3/tutorial/
4. https://www.w3schools.com/python/
5. https://www.cameralyze.co/blog/implementing-computer-vision-in-face-detection
6. https://www.ijert.org/research/face-recognition-system-IJERTV8IS050150.pdf
7. https://www.analyticsvidhya.com/blog/2022/04/object-detection-using-haar-cascade-opencv/#:~:text=Haar%20cascade%20is%20an%20algorithm,%2C%20buildings%2C%20fruits%2C%20etc.
8. https://www.mathworks.com/discovery/deep-learning.html#:~:text=Deep%20learning%20is%20a%20machine,a%20pedestrian%20from%20a%20lamppost.
9. https://en.wikipedia.org/wiki/Machine_learning
10. https://en.wikipedia.org/wiki/Computer_vision
11. https://www.ibm.com/topics/computer-vision#:~:text=Resources-,What%20is%20computer%20vision%3F,recommendations%20based%20on%20that%20information.

12. https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

13. https://www.tutorialspoint.com/python/python_gui_programming.htm#:~:text=Tkinter%20is%20the%20standard%20GUI,Tkinter%20is%20an%20easy%20task.

14. https://www.w3schools.com/MySQL/default.asp

15. https://anaconda.org/anaconda/python

GLOSSARY

1.Terminal- The terminal is an interface that lets you access the command line.

2.Algorithm- An algorithm is a procedure used for solving a problem or performing a computation.

3.Numpy- NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

4.Python Libraries- Python libraries are used to create applications and models in a variety of fields, for instance, machine learning, data science, data visualization, image and data manipulation.

5.Computer Vision- Computer vision is an interdisciplinary field that deals with how computers can be made to gain high-level understanding from digital images or videos.

6.Haar Cascade Feature- A Haar feature is essentially calculations that are performed on adjacent rectangular regions at a specific location in a detection window.

7.Tkinter- It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI.

8.Database Management System- A database management system (or DBMS) is essentially nothing more than a computerized data-keeping system.

9.MySQL- MySQL is an Oracle-backed open source relational database management system (RDBMS) based on Structured Query Language (SQL).

10.Machine Learning- Machine learning is a growing technology which enables computers to learn automatically from past data.